CS41

ACCURATE EIGENVALUES OF A SYMMETRIC TRI-DIAGONAL MATRIX

BY
W. KAHAN

TECHNICAL REPORT NO. CS41
JULY 22, 1966

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

ACCURATE EIGENVALUES OF A SYMMETRIC TRI-DIAGONAL MATRIX

By

W. Kahan*/

ABSTRACT

Having established tight bounds for the quotient of two different lub-norms of the same tri-diagonal matrix J, the author observes that these bounds could be of use in an error-analysis provided a suitable algorithm were found. Such an algorithm is exhibted, and its errors are thoroughly accounted for, including the effects of scaling, over/underflow and roundoff. A typical result is that, on a computer using rounded floating point binary arithmetic, the biggest eigenvalue of J can be computed easily to within 2.5 units in its last place, and the smaller eigenvalues will suffer absolute errors which are no larger. These results are somewhat stronger than had been known before.

---

*/ Stanford University Computer Science Department and University of Toronto Departments of Mathematics and Computer Science.
Prepared Under Contract Nonr-225(37) NR-044-211 Office of Naval Research.

Two Questions:

The following questions are connected with certain error-analyses of the computed eigenvalues of the symmetric tri-diagonal N×N matrix

$$J \equiv \begin{pmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \\ & b_2 & \cdot & \cdot & \\ & & \cdot & \cdot & b_{N-1} \\ & & & b_{N-1} & a_N \end{pmatrix} .$$

Our notation is very much like Householder's (1964); we write $\mathrm{lub}_S(A) \equiv$ (max. eigenvalue of $A^H A)^{1/2}$ and $\mathrm{lub}_E(A) \equiv \max_i \sum_j |A_{ij}| = \mathrm{lub}_E(|A|)$, where $|A|_{ij} \equiv |A_{ij}|$. The questions are

1: What bounds can be found for $\mathrm{lub}_S(J)/\mathrm{lub}_E(J)$?

2: What bounds can be found for $\mathrm{lub}_S(J)/\mathrm{lub}_S(|J|)$?

We shall see that the answers are respectively

1: $$\sqrt{\tfrac{1}{3}} \le \mathrm{lub}_S(J)/\mathrm{lub}_E(J) \le \mathrm{lub}_S(J)/\mathrm{lub}_S(|J|) .$$

2: $$\sqrt{\tfrac{1}{2}} \le \mathrm{lub}_S(J)/\mathrm{lub}_S(|J|) \le 1 .$$

The only new results here are the lower bounds $\sqrt{\frac{1}{3}}$ and $\sqrt{\frac{1}{2}}$; the other inequalities are well known and will not be proved here. (For proofs see Householder's book, §§2.2 to 2.4, with which the reader must be assumed to have extensive acquaintance.) Part of the interest in the constants $\sqrt{\frac{1}{3}}$ and $\sqrt{\frac{1}{2}}$ arises because they are best possible, and much larger

than the lower bound $\sqrt{1/N}$ which would be required if J were replaced by an arbitrary (symmetric) matrix in the inequalities above. A brief survey of such more general (and therefore weaker) bounds is given by Mrs. B. J. Stone (1962).

Proof of 1:

The results which we wish to prove are insensitive to diagonal similarity transformations and to the replacement of J by $-J$. Therefore we may assume without loss of generality that all $b_i \geq 0$. We shall write

$$b_o \equiv b_N \equiv 0 \quad \text{and}$$
$$r_i \equiv b_i + b_{i-1} .$$

Hence

$$\text{lub}_E(J) = \text{lub}_E(|J|) = \max_i(|a_i| + r_i)$$
$$= |a_k| + r_k$$

for some k defined (perhaps not uniquely) by the last equation. No generality is lost by assuming that $a_k \geq 0$, so

$$a_k + r_k = \text{lub}_E(J) \geq |a_i| + r_i \quad \text{for all} \quad i .$$

Now, $\text{lub}_S(J)$ is the largest of the magnitudes of the eigenvalues of J, so the minimax characterization of those eigenvalues (see

Householder's book, § 3.3.1) implies that

$$\mathrm{lub}_S(J) \geq \mathrm{lub}_S(K)$$

for any principal submatrix K of J. It is particularly convenient here to take

$$K \equiv \begin{pmatrix} a_{k-1} & b_{k-1} & 0 \\ b_{k-1} & a_k & b_k \\ 0 & b_k & a_{k+1} \end{pmatrix} .$$

A related matrix $\hat{K}$ is obtained by reflecting K in its skew-diagonal;

$$\hat{K} \equiv \begin{pmatrix} a_{k+1} & b_k & 0 \\ b_k & a_k & b_{k-1} \\ 0 & b_{k-1} & a_{k-1} \end{pmatrix} .$$

This reflection changes no eigenvalue, so

$$\mathrm{lub}_S(K) = \mathrm{lub}_S(\hat{K}) \geq \tfrac{1}{2}\mathrm{lub}_S(K + \hat{K}) .$$

Consequently

$$\mathrm{lub}_S(J)/\mathrm{lub}_E(J) \geq \mathrm{lub}_S(X)$$

where

$$X \equiv \tfrac{1}{2}(K + \hat{K})/(a_k + r_k) .$$

It is convenient now to define

$$x \equiv \tfrac{1}{2} r_k / (a_k + r_k) \quad \text{and}$$

$$y \equiv (a_{k-1} + a_{k+1}) / (2a_k + r_k) \quad .$$

Obviously $0 \leq x \leq \frac{1}{2}$. Also $-1 \leq y \leq 1$ because

$$\begin{aligned} |a_{k-1} + a_{k+1}| &\leq |a_{k-1}| + |a_{k+1}| \\ &\leq (a_k + r_k - r_{k-1}) + (a_k + r_k - r_{k+1}) \\ &\leq (a_k + r_k - b_{k-1}) + (a_k + r_k - b_k) \\ &= 2a_k + r_k \quad . \end{aligned}$$

The matrix X can be expressed simply in terms of x and y thus:

$$X = \begin{pmatrix} y-xy & x & 0 \\ x & 1-2x & x \\ 0 & x & y-xy \end{pmatrix} \quad .$$

A further simplification is achieved by the use of the orthogonal matrix

$$Q = \begin{pmatrix} 0 & \sqrt{\frac{1}{2}} & \sqrt{\frac{1}{2}} \\ 1 & 0 & 0 \\ 0 & \sqrt{\frac{1}{2}} & -\sqrt{\frac{1}{2}} \end{pmatrix}$$

and the 2X2 matrix

$$Z = \begin{pmatrix} 1-2x & \sqrt{2}x \\ \sqrt{2}x & y-xy \end{pmatrix} ,$$

which are connected to X by the eigenvalue-preserving similarity transformation

$$Q^T X Q = \begin{pmatrix} Z & \bigcirc \\ \bigcirc & y-xy \end{pmatrix} .$$

Since y-xy separates the two eigenvalues of Z , these two eigenvalues must be the algebraically greatest and least eigenvalues of X . Therefore our progress so far can be summarized by the inequality

$$\mathrm{lub}_S(J)/\mathrm{lub}_E(J) \geq \mathrm{lub}_S(Z) ,$$

and our result no. 1 will be proved when we have shown that

$$\mathrm{lub}_S(Z) \geq \sqrt{\tfrac{1}{3}} .$$

This last inequality is obtained below from a demonstration that

$$\sqrt{\tfrac{1}{3}} = \min.\mathrm{lub}_S(Z) \quad \text{over} \quad (0 \leq x \leq \tfrac{1}{2} \quad \text{and} \quad -1 \leq y \leq 1) .$$

Let the eigenvalues of Z be regarded now as functions of y for a fixed x . They are both monotonic non-decreasing functions of y because

any increase in y is tantamount to adding to Z some positive multiple of the positive semi-definite matrix

$$\begin{pmatrix} 0 & 0 \\ 0 & 1-x \end{pmatrix} .$$

The value

$$y_o \equiv -(1 - 2x)/(1 - x)$$

satisfies $-1 \leq y_o \leq 0$; and when $y = y_o$ the eigenvalues of Z are just $+z_o$ and $-z_o$, where

$$z_o \equiv \sqrt{6x^2 - 4x + 1} \ .$$

The values y_o and z_o are significant because for any other value of y the matrix Z has either a positive eigenvalue $\geq z_o$

or a negative eigenvalue $\leq -z_o$.

In other words,

$$z_o = \min \operatorname{lub}_S(Z) \quad \text{over} \quad -1 \leq y \leq 1$$

for any fixed x in $0 \leq x \leq \frac{1}{2}$. And z_o's minimum value $\sqrt{\frac{1}{3}}$ is taken when $x = \frac{1}{3}$.

The foregoing proof that $\operatorname{lub}_S(J)/\operatorname{lub}_E(J) \geq \sqrt{\frac{1}{3}}$ also points to an example

$$J = \begin{pmatrix} -1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & -1 \end{pmatrix}$$

with $\mathrm{lub}_S(J) = \sqrt{3}$ and $\mathrm{lub}_E(J) = 3$; therefore the lower bound $\sqrt{\frac{1}{3}}$ cannot be increased.

Proof of 2:

We wish to show that $\mathrm{lub}_S(J)/\mathrm{lub}_S(|J|) \geq \sqrt{\frac{1}{2}}$. As before, we assume without loss of generality that all $b_i \geq 0$. It is convenient to begin with some definitions. First let

$$\mu \equiv \mathrm{lub}_S(|J|)$$

$$= \text{max. eigenvalue of } |J| \ .$$

Second, define

$$M \equiv \tfrac{1}{2}(|J| - J) \ .$$

Evidently M is a non-negative diagonal matrix whose positive elements are just the positive diagonal elements of $-J$. For the sake of symmetry we should like to have a similar definition for the non-negative diagonal matrix P whose positive elements are just the positive diagonal elements of $+J$. Such a definition is provided in stages as follows. We define

$$E \equiv \mathrm{diag}(-1, +1, -1, \ldots, (-1)^N) \ ,$$

and use it in an eigenvalue-preserving similarity transformation to define the matrix

$$\bar{J} = -EJE$$

whose eigenvalues and diagonal elements are just the negatives of those of J. But $\bar{J}$ has the same off-diagonal elements as J and $|J|$. Therefore the matrix

$$P \equiv \frac{1}{2}(|J| - \bar{J})$$

is defined in much the same way as was M. Note that $PM = 0$. Finally, because J^2 and $\bar{J}^2$ have the same eigenvalues,

$$\lambda \equiv \text{lub}_S(J) = \text{lub}_S(\bar{J}) \quad .$$

Now we may proceed to demonstrate that $\lambda/\mu \geq \sqrt{\frac{1}{2}}$.

According to the theory of non-negative matrices outlined in §2.4 of Householder's book, there must exist some non-negative vector $\underline{v}$ such that

$$|J|\underline{v} = \mu\underline{v} \geq 0 \quad \text{and} \quad \underline{v}^T\underline{v} = 1 \quad .$$

Since $(\text{lub}_S(J))^2 = \max . \underline{x}^T J^2 \underline{x}$ over $\underline{x}^T\underline{x} = 1$,

$$\lambda^2 \geq \underline{v}^T J^2 \underline{v} = \underline{v}^T(|J| - 2M)^2\underline{v}$$

$$= \mu^2 - 4\mu\underline{v}^T M\underline{v} + 4\underline{v}^T M^2\underline{v} \quad .$$

Similarly,

$$\lambda^2 \geq \underline{v}^T \bar{J}^2 \underline{v} = \underline{v}^T (|J| - 2P)^2 \underline{v}$$

$$= \mu^2 - 4\mu \underline{v}^T P \underline{v} + 4\underline{v}^T P^2 \underline{v} \quad .$$

Adding and using the fact that $PM = 0$ yields

$$2\lambda^2 \geq 2\mu^2 - 4\underline{v}^T (M + P)(\mu I - M - P)\underline{v} \quad .$$

But $$M + P = \text{diag}(|a_i|) \quad , \quad \text{and}$$

$$|a_i|(\mu - |a_i|) \leq \mu^2/4 \quad .$$

Therefore

$$4\underline{v}^T (M + P)(\mu I - M - P)\underline{v} \leq \mu^2 \underline{v}^T \underline{v} = \mu^2$$

and so

$$2\lambda^2 \geq 2\mu^2 - \mu^2 = \mu^2$$

as desired. Result no. 2 is proved.

This proof points less directly than did the proof of result no. 1 to an example J for which the second bound is achieved, i.e. for which

$$\text{lub}_S(J) = \sqrt{\frac{1}{2}}\,\text{lub}_S(|J|) \quad .$$

In fact, the foregoing proof was motivated by a foreknowledge of the following example.

Let $a_i = \frac{1}{2}(-1)^i x$ for $1 \le i \le N$, and $b_i = \frac{1}{2}$ for $1 \le i < N$. The value of x will be chosen later to be the same as μ defined above, but first we observe that now

$$P + M = \tfrac{1}{2}xI \ , \ P - M = \tfrac{1}{2}xE \ ,$$

and

$$C \equiv |J| - \tfrac{1}{2}xI = J - \tfrac{1}{2}xE$$

is an $N \times N$ matrix with zero on the diagonal and $\frac{1}{2}$ on the subdiagonal and superdiagonal. The eigenvalues of C are well-known; they are just the numbers

$$\gamma_n \equiv \cos n\pi/(N + 1) \quad \text{for} \quad n = 1,2,\ldots, N \ .$$

(See Householder's book, p. 34 ex. 50. His matrix J is defined on p. 2 and differs from ours. His $K = 2C$.) In particular, since $|J| = C + \frac{1}{2}xI$,

$$\mu = \gamma_1 + \tfrac{1}{2}x \ .$$

Next let us compute the largest eigenvalue λ^2 of J^2. The computation is considerably shortened by Jim Varah's observation that

$$J^2 = C^2 + \tfrac{1}{4}x^2 I \ .$$

Therefore $\lambda^2 = \gamma_1^2 + \frac{1}{4}x^2$. The ratio λ/μ takes on its minimum value $\sqrt{\frac{1}{2}}$ when $x = \mu = 2\gamma_1$. This example shows that the lower bound $\sqrt{\frac{1}{2}}$ cannot be increased.

Application:

Let the eigenvalues λ_i of J be ordered thus:

$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{N-1} \leq \lambda_N \ ;$$

and suppose the eigenvalues $(\lambda_i + \delta\lambda_i)$ of $(J + \delta J)$ are ordered similarly; $\lambda_i + \delta\lambda_i \leq \lambda_{i+1} + \delta\lambda_{i+1}$. Here the matrix δJ is a perturbation attributed, possibly, to rounding errors in a numerical calculation. We shall assume that δJ is tri-diagonal with elements bounded by, say,

$$|\delta a_i| \leq \alpha |a_i| \quad \text{and} \quad |\delta b_i| \leq \beta |b_j|$$

where α and β are small positive constants. Given α and β , how big can $\delta\lambda_i$ be ?

The easiest bound for $\delta\lambda_i$ uses the fact that, if the eigenvalues of δJ are

$$\delta_1 \leq \delta_2 \leq \cdots \leq \delta_N \ ,$$

then $$\delta_1 \leq \delta\lambda_i \leq \delta_N \ .$$

A proof of this relation can be found in Householder's book (1964) p. 79. Consequently

$$|\delta\lambda_i| \leq \text{lub}_S(\delta J) \leq \text{lub}_S(|\delta J|) \leq \text{lub}_E(\delta J) \quad .$$

In particular, if $\alpha = \beta$ then $|\delta J| \leq \alpha|J|$ elementwise, so

$$|\delta\lambda_i| \leq \sqrt{2}\alpha \ \text{lub}_S(J) = \sqrt{2}\alpha \ \max_j|\lambda_j|$$

for all i by virtue of the inequality no. 2. More generally, inequality no. 2 can be extended without any difficulty to the case that $\alpha \neq \beta$ and yields the bound

$$|\delta\lambda_i| \leq \text{lub}_S(|\delta J|) \leq \sqrt{\alpha^2 + \beta^2}\ \text{lub}_S(J) \quad .$$

Though pessimistic, these bounds are slightly stronger than the best bounds available in terms of $\text{lub}_E(J)$. But are there any practical circumstances where such bounds may be of use? They rely upon the inequalities

$$|\delta a_i| \leq \alpha|a_i| \quad \text{and} \quad |\delta b_i| \leq \beta|b_i| \quad ,$$

whereas the typical rounding error analyses of the past have contained weaker constraints like

$$|\delta a_i| \leq \alpha \ \text{lub}_S(J) \quad \text{and} \quad |\delta b_i| \leq \beta \ \text{lub}_S(J)$$

(cf. Wilkinson's book (1965) p. 304). Thus we are faced with the following problem:

Given a set of error-bounds, find a numerical algorithm to which they are applicable.

This problem has an elegant solution which is described below.

The Algorithm:

We shall now exhibit and completely error-analyze a simple and effective method for computing any eigenvalue λ_k of J. The basic method was first put forth in Dr. Boris Davison's numerical analysis lectures at the University of Toronto in 1959, and begins with

Sylvester's Law of Inertia:

Suppose $A = A^T$ is symmetric

L is non-singular, and

$D \equiv L^{-1}A(L^{-1})^T$ is diagonal.

Then the numbers of positive, negative and zero diagonal elements of D are the same respectively as the numbers of positive, negative and zero eigenvalues of A.

A proof may be found in any standard text on matrices; e.g. in Gantmacher (1959) vol. I p. 297. We shall apply this Law to the triangular factorization of

$$J - xI = LU = LDL^T$$

into triangular bi-diagonal matrices L and U obtained by Gaussian elimination without pivotal interchanges. It is unnecessary to compute any but the diagonal elements u_n of U. They are obtained from the simple recurrence

$$u_1 = a_1 - x \qquad \text{and}$$
$$u_n = a_n - x - b_{n-1}^2/u_{n-1} \quad \text{for} \quad n = 2,3,\ldots, N \ .$$

This recurrence breaks down if and only if some value $u_n = 0$, but such a thing can happen only if x takes on one of at most $\frac{1}{2}N(N+1)$ exceptional values. Indeed, it is easy to see that

$$u_n = u_n(x) = \varphi_n(x)/\varphi_{n-1}(x)$$

where $\varphi_n(x)$ is the characteristic polynomial of the first $n \times n$ principal submatrix of J. In particular,

$$\varphi_N(x) = \det(J - xI) \quad .$$

Consequently, the recurrence can break down only if x coincides with one of the eigenvalues of one of the leading principal submatrices of J. Let us postpone the discussion of these exceptional values of x; suppose for now that the recurrence is successful, and compute

$$\nu(x) \equiv (\text{the number of values } u_n(x) < 0) \quad .$$

Sylvester's Law implies that

$$\nu(x) = (\text{the number of } J\text{'s eigenvalues } \lambda_i < x) \quad .$$

Therefore any selected eigenvalue λ_k can be computed as the limit of a sequence of nested intervals $[\underline{x}_m, \overline{x}_m]$ with

$$\nu(\underline{x}_m) < k \leq \nu(\overline{x}_m) \quad \text{for all } m \ ,$$

$$\underline{x}_m \le \underline{x}_{m+1} \le \bar{x}_{m+1} \le \bar{x}_m \quad \text{for all} \quad m \ ,$$

and
$$\bar{x}_m - \underline{x}_m \to 0 \quad \text{as} \quad m \to \infty \ .$$

The mechanism by which the successive values $\underline{x}_m$ and $\bar{x}_m$ are chosen is of no consequence here; a bisection method could be used (cf. Wilkinson (1962)), though that is slow. A faster algorithm has been produced by the author and Jim Varah (1966). But the error-analysis is independent of the way in which the values $\underline{x}_m$ and $\bar{x}_m$ are chosen provided they have the properties listed above.

So far we have not seen anything very new. Indeed, the function $\nu(x)$ is just the number of variations of sign in the Sturm sequence

$$\varphi_o \equiv 1 \ , \ \varphi_1(x) \ , \ \varphi_2(x) \ , \ldots , \ \varphi_N(x)$$

which has been in use for over a decade to compute the eigenvalues of symmetric tri-diagonal matrices. (See Wilkinson's book (1965) p. 299-312. Also see Householder's book (1964) p. 86-7 ex. 10 and 11, and p. 175 ex. 14; his φ_n differs from ours by a factor of $(-1)^n$.) However, the φ-recurrence

$$\varphi_n = (a_n - x)\varphi_{n-1} - b_{n-1}^2\varphi_{n-2}$$

takes more time on most machines than does the u-recurrence; and over/underflow is an inescapable complication in the φ-recurrence whereas the u-recurrence can be rendered almost immune to over/underflow. These are the reasons Davison gave for his preference of the u-recurrence. Unfortunately, he died before he had the chance to show how well behaved his method could be. The task of analysis is now ours.

Over/Underflow on the machine

Over/underflow in the u-recurrence can easily be rendered insignificant by a proper preliminary scaling of the data a_i and b_i. The description of the scaling process begins with a definition of certain machine constants:

Ω is the greatest floating point number normally represented directly in the machine.

η is the smallest positive (non-zero) floating point number normally represented directly.

ϵ is the smallest positive floating point number such that the computed value of $1.0 + \epsilon$ differs from 1.0 after it is rounded or truncated to the precision being carried.

The following table lists typical values for these parameters:

Machine	Ω	η	ϵ single precision	ϵ double precision *	$\tau = \eta^{1/4}\Omega^{-1/2}$
IBM 7094	$< 2^{127} = 1.7\times10^{38}$	$2^{-129} = .15\times10^{-38}$	truncated: $2^{-26} = 1.5\times10^{-8}$ rounded: $2^{-27} = .75\times10^{-8}$	$2^{-53} = 1.1\times10^{-16}$	1.5×10^{-29}
Burroughs B5500	$< 8^{76} = 4.3\times10^{68}$	$8^{-51} = .9\times10^{-47}$	$\frac{1}{2}8^{-12} = .73\times10^{-11}$	$8^{-25} = .377\times10^{-22}$	$.8\times10^{-46}$
IBM 360 series	$< 16^{63} = 7.2\times10^{78}$	$16^{-65} = .54\times10^{-78}$	$16^{-5} = .95\times10^{-6}$	$16^{-13} = 2.2\times10^{-16}$	10^{-59}
G.E. 635	$< 2^{127} = 1.7\times10^{38}$	$2^{-129} = .15\times10^{-38}$	$2^{-26} = 1.5\times10^{-8}$	$2^{-62} = .46\times10^{-18}$	1.5×10^{-29}
Control Data 3600	$< 2^{1023} = .9\times10^{308}$	$2^{-1024} = .56\times10^{-308}$	$2^{-36} = 1.4\times10^{-11}$	$2^{-84} = .19\times10^{-25}$	$.9\times10^{-231}$
Control Data 6600	$< 2^{1023} = .9\times10^{308}$	$2^{-1024} = .56\times10^{-308}$	$2^{-48} = 3.5\times10^{-15}$	$2^{-96} = 1.1\times10^{-29}$	$.9\times10^{-231}$

TABLE OF MACHINE-DEPENDENT CONSTANTS

* Only a few machines adhere to this precision in all double precision operations.

We must also make certain assumptions about the treatment of arithmetic over/underflows, because they will occur. First, we assume that whenever any floating point arithmetic operation (+ , - , × , /) underflows its result will be cleared to zero. Second, we assume that whenever any operation overflows its result will be set to $+\Omega$ or $-\Omega$ with the correct sign. The preservation of sign after overflow is essential. Fortunately, these conventions for the treatment of over/underflow are widely used on many machines, including the IBM 7094 and, possibly, the Burroughs B5500. Unfortunately, the new IBM 360 series hardware forgets the sign after overflow, but presumably that oversight will soon be corrected. It is possible to prevent the u-recurrence from overflowing at all, but to do so costs a noticeable retardation on most computers, as we shall see.

If over/underflow is treated as described above, any over/underflow occurring in the u-recurrence will be practically inconsequential for reasons to be given later. Therefore we must make a third assumption; we assume that the program can inhibit the production of diagnostic over/underflow messages and can ignore any over/underflow indicators that might otherwise serve as superfluous distractions during the computation of the u's . (This is not meant to imply that those indicators are superfluous in any other context. Quite the contrary!)

If all three assumptions about the treatment of over/underflow are valid then they cope with the problem far more simply, elegantly and economically than any other scheme known to the author. There is reason to doubt that any comparable scheme could ever be devised for the φ-recurrence.

Scaling:

Now let the scale factor σ be defined as the largest power of the machine's arithmetic base which satisfies

$$\sigma|a_i| \le \tau\Omega \quad \text{and} \quad \sigma|b_i| \le \tau\Omega \quad \text{for all} \quad i \ ,$$

where

$$\tau = \eta^{1/4}\Omega^{-1/2} \ .$$

The significance of this constant τ is that over/underflow will later be shown to contribute an absolute error no larger in magnitude than about $4\tau \cdot \mathrm{lub}_S(J)$ to the computed eigenvalues. The values of τ tabulated above show how small τ usually is compared with the rounding error level ϵ. Evidently over/underflow will hardly ever restrict the range of magnitudes spanned by the accurately computed eigenvalues of J nearly as much as do rounding errors.

Normally σ is approximately

$$\tau\Omega/\mathrm{max.}(\mathrm{max}_i|a_i| \ , \ \mathrm{max}_i|b_i|) \ ;$$

but there are exceptional cases where that expression would overflow, so σ must be set instead to the largest power of the machine's arithmetic base. These cases are ignored in what follows because they are susceptible to a simpler analysis with the same results as are demonstrated below.

After σ is known, the matrix J is scaled by being replaced by σJ. Since σ is a power of the base there are no rounding errors. But underflows may occur. These underflows result in the annihilation of at most

those elements a_i and b_i which satisfy

$$|a_i| < \eta^{1/2}\tau \ \mathrm{lub}_S(J) \text{ or } |b_i| < \eta^{1/2}\tau \ \mathrm{lub}_S(J) \ .$$

These perturbations are negligible compared with what follows, so they may be ignored. Later the computed eigenvalues λ_i will be unscaled by dividing them all by σ. Any over/underflow which occurs here is fully deserved and must be reported by the diagnostic machinery mentioned above to indicate that some eigenvalues (just the ones that over/underflow) cannot be represented in the normal way without over/underflow. Nothing more need be said about scaling: we merely assume henceforth that

$$\tau\Omega \leq \mathrm{lub}_S(J) \leq 3\tau\Omega \ .$$

Two programs:

Now is the time to write out the u-recurrence explicitly in, say, FORTRAN. There are two versions, according as overflow is prevented or allowed. Both versions begin by constructing the arrays BB and A containing

$$\left.\begin{aligned} BB(I) &= b_{I-1}^2 \\ A(I) &= a_I \end{aligned}\right\} \quad \text{for} \quad I = 1,2,\ldots, N \ .$$

If $|b_{I-1}| < \sqrt{\eta}$ then BB(I) will underflow to zero, but this amounts to a perturbation of no more than

$$\sqrt{\eta} = \tau \cdot (\tau\Omega) \leq \tau \cdot \mathrm{lub}_S(J)$$

in the given matrix J, and is included in the error analysis given later. Note that BB(1) = 0.0 by definition, and that we can assume that

$$|X| \leq \mathrm{lub}_E(J) \leq 3\tau\Omega$$

is satisfied by any number X which might usefully be considered as an estimate of an eigenvalue.

Here is the segment of code which prevents any overflow in the u-recurrence; the constant RTETA has the value

$$\mathrm{RTETA} = \sqrt{\eta} \ .$$

The FORTRAN symbol ".GT." stands for ">" .

```
              U = 1.0
             NU = 0
             DO 3    I = 1,N
                U = (A(I) - BB(I)/U) - X
                IF (U .GT. RTETA) GO TO 3
1               IF (U .GT. - RTETA) U = - RTETA
2               NU = NU + 1          ... when U < 0  .
3               CONTINUE
           ... Now NU = ν(X) ...
```

Whenever the computed value of u_I lies between $-\sqrt{\eta}$ and $+\sqrt{\eta}$, it is replaced in statement 1 by $-\sqrt{\eta}$. Consequently the quotient b_I^2/u_I never exceeds

$$(\tau\Omega)^2/\sqrt{\eta} = \Omega \quad ,$$

so overflow is impossible. Of course, u_I may have been decreased in statement 1 by as much as $2\sqrt{\eta}$, but this too is no larger than might have been caused by decreasing a_I by the allowable perturbation

$$2\sqrt{\eta} \leq 2\tau \cdot \mathrm{lub}_S(J) \quad .$$

Here is a simpler and faster program segment which is useable whenever overflow is treated according to the conventions described above. The constant ETA = η .

```
                U = 1.0
               NU = 0
               DO  3     I = 1,N
                  U = (A(I) - BB(I)/U) - X
                 IF (U)   2, 1, 3
1                   U = - ETA      ... if U was 0 .
2                   NU = NU + 1    ... if U was ≤ 0 .
3                CONTINUE
              ... Now  NU = ν(X) ...
```

Note that whenever any u_I vanishes it is replaced in statement 1 by $u_I = -\eta$ to forestall a subsequent division by zero. Whenever (rarely) any u_I overflows, its sign remains unchanged so that NU is treated correctly; then u_{I+1} is in error because the computed value of b_I^2/u_I must be larger in magnitude than it should be. But the error in u_{I+1} is no worse than might have been caused by perturbing a_{I+1} by at worst

$$(\tau\Omega)^2/\Omega = \tau(\tau\Omega) \leq \tau \cdot \mathrm{lub}_S(J) \quad .$$

And underflow, if it occurs, causes no more perturbation than η, which is negligible.

All told, these subterfuges for circumventing the ill effects of over/underflow cause the computed value NU to be, instead of $\nu(X)$, some value that would have been obtained had J first been changed in each element by at most

$$2\tau \cdot \mathrm{lub}_S(J) \quad \text{(in the first program)} \quad \text{or}$$

$\tau \cdot \mathrm{lub}_S(J)$ (in the second program)

before $\nu(X)$ was computed without any intervention on behalf of over/underflow. These perturbations will be shown later to affect the computed eigenvalues by no more than $4\tau \cdot \mathrm{lub}_S(J)$. First we should consider one last programming detail.

The k^{th} eigenvalue λ_k is the k^{th} jump-point of the integer valued function $\nu(x)$;

$$\lim_{x \to \lambda_k^-} \nu(x) < k \leq \lim_{x \to \lambda_k^+} \nu(x) \quad .$$

(The multiplicity of the jump-point λ_k is just the difference between the upper and lower limits.) Can a similar statement be made about the computed approximation NU(X) ? If so, any algorithm that works properly for the exact function $\nu(X)$ will work properly for its approximation NU(X) . If not, if NU(X) could have more than N jump-points, then great care would be required to design the algorithm in such a way that it could not be confused by spurious jumps down. As it happens, no such care is required on most machines.

We shall demonstrate below that, despite rounding errors and over/underflow, the computed function NU(X) is a monotonic non-decreasing integer valued function of X with just N jumps. The only assumption is that each arithmetic operation executed by the machine is a monotonic function of its arguments despite rounding. For example, if A , B and C are all positive numbers represented in the machine, and if the FORTRAN program

```
X1 =  A + B

Y1 = (A + C) + B

X2 =  A - B

Y2 = (A + C) - B

X3 =  A * B

Y3 = (A + C) * B

X4 =  A / B

Y4 = (A + C) / B

X5 =  B / (A + C)

Y5 =  B / A
```

is executed, then $XI \leq YI$ for all $I = 1,2,3,4$ or 5. This assumption is certainly valid for single precision computations on all of the machines listed in the table above. Indeed, the builder of any machine which failed to satisfy this assumption should be ashamed of himself.

The Monotonicity of NU(X):

The monotonicity of NU(X) will be derived as a consequence of the properties of the successive values of U, for which some notation is required. Given any argument X, a number representable in the machine, either program above will produce a sequence of values $U_n(X)$ and $NU_n(X)$, the values taken by U and NU respectively after statement 3 has been executed for the n^{th} time. In particular,

$$U_o(X) \equiv 1.0 \text{ and } NU_o(X) \equiv 0 \ ;$$
$$U_1(X) = [A(1) - X] \text{ rounded etc. and}$$
$$NU_1(X) = 0 \text{ if } X \leq A(1) - \theta$$
$$= 1 \text{ if } X > A(1) - \theta \ ,$$

where $\theta = \sqrt{\eta}$ in the first program
$= \eta$ in the second program .

Note that no $U_n(X)$ can lie closer to zero than $+\theta$ or $-\theta$. At the end of the DO-loop,

$$U = U_N(X) \text{ and } NU = NU(X) = NU_N(X) \ .$$

The interesting values of X are those where some $U_n(X)$ changes sign. These points shall be identified precisely with the aid of a notation X' for the successor of X; if X is a number representable in the machine and eligible to be an argument for the programs above, then X' is the next larger eligible argument. Normally X' will exceed X by one

unit in their last place being carried in the computation.

A "zero" Z of $U_n(X)$ is now defined to be any argument Z which satisfies both

$$U_n(Z) \geq \theta \quad \text{and} \quad -\theta \geq U_n(Z') \quad .$$

A "pole" Y of $U_n(X)$ is any argument Y which satisfies

$$U_n(Y) < U_n(Y') \quad .$$

$U_n(X)$ can change sign only at a zero or a pole, though $U_n(X)$ may fail to change sign at some poles. Between any two zeros of $U_n(X)$ must lie at least one pole where $U_n(X)$ changes sign, and possibly some other poles where $U_n(X)$ does not change sign. Let us examine these poles more closely.

If Y is a pole of $U_n(X)$ then

$$\lceil BB(n)/U_{n-1}(Y)] > [BB(n)/U_{n-1}(Y')]$$

because the contrary relation would prevent

$$U_n(X) = [[A(n) - [BB(n)/U_{n-1}(X)]] - X] \quad ,$$

where each pair of brackets means

"round [...] and take care of over/underflow, if any" ,

from increasing when X moves from Y to Y'. (Note that the over/underflow subterfuges do not destroy the monotonicity of the arithmetic operations even if $U_n(X)$ has to be replaced by $-\Theta$.) Therefore either Y is a pole of $U_{n-1}(X)$ where U_{n-1} does not change sign, or Y is a zero of $U_{n-1}(X)$. A backward induction yields the following statement:

> If Y is a pole of $U_n(X)$, then there exists some positive integer $m < n$ such that
>
> $$U_m(Y) > 0 > U_m(Y') \quad ,$$
>
> and for all integers i (if any) strictly between m and n
>
> $$U_i(Y) < U_i(Y')$$
>
> with no change of sign.

We observe that $U_1(X)$ has no poles and just one zero. Therefore, as $U_2(X)$ is carried from $U_2(-\Omega) = \Omega$ to $U_2(\Omega) = -\Omega$, it can have at most two zeros separated by one pole where U_2 changes sign, or one zero and one pole where U_2 does not change sign, or one zero and no poles if $BB(2)$ is very tiny. In all cases one can verify with ease that $NU_2(X)$ is a monotonic non-decreasing function of X with at most two distinct jumps from $NU_2(-\Omega) = 0$ to $NU_2(\Omega) = 2$. Rather than extend this desired property to $NU_n(X)$ for all n by a long constructive argument, we shall show that a failure of $NU_n(X)$ to be monotonic would create a contradiction.

Let n be the smallest integer for which $NU_n(X)$ is not a monotonic function. Obviously $n > 1$. Suppose $NU_n(X)$ fails to be monotonic at Y; since

$$0 = NU_n(-\Omega) \leq NU_n(X) \leq n = NU_n(\Omega) \quad ,$$

the failure must take the form

$$NU_n(Y) > NU_n(Y') \quad .$$

However, our hypothesis about n implies that $NU_{n-1}(X)$ is monotonic, which means

$$NU_{n-1}(Y) \leq NU_{n-1}(Y') \quad .$$

Also,

$$NU_n(X) - NU_{n-1}(X) = 0 \quad \text{if} \quad U_n(X) > 0$$

$$= 1 \quad \text{if} \quad U_n(X) < 0 \quad ,$$

so

$$0 > NU_n(Y') - NU_n(Y)$$

$$= [NU_n(Y') - NU_{n-1}(Y')]$$

$$+ [NU_{n-1}(Y') - NU_{n-1}(Y)]$$

$$+ [NU_{n-1}(Y) - NU_n(Y)]$$

$$\geq [0] + [0] + [-1] = -1 \quad .$$

But this implies, {term} by {term} , that

$$NU_n(Y') = NU_{n-1}(Y') \quad \text{and} \quad U_n(Y') > 0 \ ,$$

$$NU_{n-1}(Y') - NU_{n-1}(Y) = 0 \ , \quad \text{and}$$

$$NU_n(Y) = NU_{n-1}(Y) + 1 \quad \text{and} \quad U_n(Y) < 0 \ .$$

Evidently Y is a pole of $U_n(X)$. Therefore there exists some positive integer $m < n$ for which Y is a zero of $U_m(X)$; we shall have

$$U_m(Y) > 0 > U_m(Y') \ .$$

Therefore
$$\begin{aligned} NU_m(Y') &= NU_{m-1}(Y') + 1 \quad \text{because} \quad U_m(Y') < 0 \ , \\ &\geq NU_{m-1}(Y) + 1 \quad \text{by monotonicity} \ , \\ &= NU_m(Y) + 1 \quad \text{because} \quad U_m(Y) > 0 \ . \end{aligned}$$

Also, if there are any integers i strictly between m and $n-1$,

$$NU_i(Y') - NU_{i-1}(Y') = NU_i(Y) - NU_{i-1}(Y)$$

because Y is a pole of U_i with no change in sign. Therefore

$$NU_{n-1}(Y') - NU_{n-1}(Y) = NU_m(Y') - NU_m(Y) \geq 1 \ ,$$

whereas we saw above that

$$NU_{n-1}(Y') - NU_{n-1}(Y) = 0 \ .$$

This contradiction proves that NU(X) is a monotonic non-decreasing function of X, as desired.

It seems surprising that so strong a result can be proved with no appeal to the continuum, nor any estimate for the errors in the values $U_n(X)$. On the contrary, the values of $U_n(X)$ can be completely different from the mathematically exact values u_n that would have been obtained without rounding errors nor over/underflow, even to the extent of having the wrong signs. Fortunately, the errors in the intermediate results $U_n(X)$ are of no interest beyond an assurance that the errors are not haphazard. And the behaviour of NU(X) provides just that assurance.

Bounding Rounding Errors:

The next step is to show that if $|X| \leq \text{lub}_E(J)$ then $NU(X)$ is precisely the value that $\nu(X)$ would have taken if J had been replaced by some nearby matrix $J(X)$ and all computations had been carried out infinitely precisely with neither rounding errors nor over/underflow subterfuges. The principles behind the analysis that follows are very much like those to be found in Wilkinson's books (1963, 1965). We shall try to describe the elements of $J(X)$ in terms of the numbers that actually appear in the arithmetic registers of the machine during the computation, and in terms of the rounding error bound ϵ tabulated above for several machines. The ideas involved are best illustrated by the following examples.

The FORTRAN assignment statement

$$C = A * B$$

will not replace C by the product of A and B, but will instead set C to a value

$$C = (1 + \gamma)AB$$

in which $|\gamma|$ is normally bounded by, say,

$$|\gamma| < \epsilon \quad .$$

Note that A, B and the new value C are defined quite precisely, and satisfy the previous equation exactly. The only unknown quantity is γ,

but $|\gamma|$ is bounded by a known value ϵ except when over/underflow intervenes. Similarly, the assignment statement

$$C = A/B$$

actually stores a value

$$C = (1 + \beta)A/B$$

where $$|\beta| < \epsilon \ .$$

As a matter of fact, the situation is not always as described above. In double-precision the values of β and γ can be as large as 3ϵ on a 7094, 5ϵ on a B5500, and 16ϵ on a 360. These unnecessarily large errors are so repugnant to the author that he takes the liberty of passing them directly from the machines' manufacturers to the reader, who may accommodate them by multiplying ϵ in the bounds given below by whichever factor is appropriate for his machine. For a similar reason, the author chooses to presume that the FORTRAN statement

$$C = A + B$$

causes C to take precisely the value

$$C = (A + B)/(1 + \alpha)$$

with $|\alpha| < \epsilon$. Only in double-precision, and then only on some machines, is it necessary to replace the last two relations by

$$C = (1 + \alpha)A + (1 + \beta)B$$

with $|\alpha| < \epsilon$ and $|\beta| < \epsilon$, but this will weaken the error bounds to be given below by a factor no larger than two.

The Construction of J(X) near J:

The first step in the construction of $J(X)$ is the definition of certain values A_I and B_I corresponding respectively to a_I and b_I. Let us set

$$B_I = b_I \quad \text{if} \quad B_I^2 \text{ does not underflow} ,$$

$$= 0 \quad \text{if} \quad |b_I| < \sqrt{\eta} .$$

In either event, $|B_I - b_I| < \sqrt{\eta}$. And

$$BB_I = BB(I) = (1 + \beta_I')B_{I-1}^2 ,$$

where β_I' is the relative error due to multiplication and satisfies $|\beta_I'| < \epsilon$. (The primes used during the construction of $J(X)$ do not denote successors.)

The value of A_I depends upon X, and differs from a_I only to the extent required to compensate for the effects of over/underflow. Reasons

have already been given why we should expect that

$$|A_I - a_I| \le 2\sqrt{\eta} \text{ in the first program ,}$$

$$\le \sqrt{\eta} \text{ in the second program ,}$$

execpt for an ignorable contribution no larger than η . Let us stay with the second program from now on, and ignore not only the scaling underflow error η by setting

$$A(I) \equiv a_I ,$$

but also agree to ignore the comparable error induced by underflow or statement 1 during the computation of U_I .

The dissection of the FORTRAN statement

$$U = ((A(I) - BB(I)/U) - X)$$

to find its rounding errors is an inductive process. For $I = 1$ we define

$$A_1 \equiv A(1) = a_1 \qquad \text{and}$$

$$v_1 \equiv A_1 - X \qquad \text{precisely, and}$$

$$U_1 = [[A(1) - 0.] - X] = v_1/(1 + \alpha_1'') ,$$

where $|\alpha_1''| \le \epsilon$. Evidently $\text{sign}(v_1) = \text{sign}(U_1)$.

We also set $\alpha'_1 = \beta''_1 = 0$ and $v_0 = 1$. The induction hypothesis is that for $n = 1,2,\ldots, I-1$ we may write

$$v_n = A_n - (1 + \beta''_n)(1 + \alpha''_{n-1})(1 + \alpha'_{n-1})BB_n/v_{n-1} - (1 + \alpha'_n)X \ ,$$

$$|A_n - a_n| \leq \sqrt{\eta} \ ,$$

$$|\beta''_n| < \epsilon \ , \ |\alpha'_n| < \epsilon \ , \ |\alpha''_n| < \epsilon \ , \qquad \text{and}$$

either $v_n = (1 + \alpha''_n)(1 + \alpha'_n)U_n$ or

$$|U_n| = \Omega \ \text{ and } \ 0 < U_n/v_n < 1 \ \text{ and } \ \alpha'_n = \alpha''_n = 0 \ .$$

In any case, $\text{sign}(v_n) = \text{sign}(U_n) \neq 0$.

The hypothesis is obviously true for $n = 1$, since $BB_1 = 0$. Note that U_n represents what was earlier referred to as $U_n(X)$, and is a number actually stored in the computer. The values BB_n and a_n are also stored in the computer, but A_n will not be stored if it differs from a_n, and v_n is a figment of the imagination except for its sign.

Now for the advance to $n = I$. The first value to be inspected is

$$[BB(I)/U_{I-1}]\text{rounded} = (1 + \beta''_I)BB_I/U_{I-1} \ ,$$

where $|\beta''_I| < \epsilon$ unless overflow occurs. (Underflow is being ignored.) If this quotient overflows then the remaining arithmetic operations are irrelevant because the scaling has ensured that neither A(I) nor X can be bigger than

$$3\tau\Omega \ll \epsilon\Omega \ ;$$

therefore overflow will cause U_I to be given the value

$$U_I = -\Omega \ \mathrm{sign}(U_{I-1}) \ ,$$

and we may define

$$A_I \equiv a_I \ , \ \beta''_I \equiv \alpha'_I \equiv \alpha''_I \equiv 0 \ , \quad \text{and}$$

$$v_I \equiv (A_I - BB_I/U_{I-1}) - X \ .$$

These values satisfy the induction hypothesis.

If the quotient $[BB_I/U_{I-1}]$ does not overflow, there is still the possibility that the previous quotient $[BB_{I-1}/U_{I-2}]$ overflowed to be considered. In this case $|U_{I-1}| = \Omega$ and $0 < U_{I-1}/v_{I-1} < 1$, and we define

$$A_I \equiv A(I) - [BB(I)/U_{I-1}] + (1 + \beta''_I)BB_I/v_{I-1} \ .$$

Evidently

$$|A_I - a_I| = (1 + \beta''_I)BB_I|1/U_{I-1} - 1/v_{I-1}|$$

$$< (1 + \epsilon)(\tau\Omega)^2|1/\Omega| = (1 + \epsilon)\sqrt{\eta} \ .$$

The factor $1 + \epsilon$ is unimportant and shall be dropped. Note too that

$$A(I) - [BB(I)/U_{I-1}] = A_I - (1 + \beta''_I)BB_I/v_{I-1}$$

$$= A_I - (1 + \beta''_I)(1 + \alpha''_{I-1})(1 + \alpha'_{I-1})BB_I/v_{I-1} \ .$$

The last relation is satisfied too if neither $[BB(I)/U_{I-1}]$ nor $[BB(I-1)/U_{I-2}]$ overflowed, in which case we set $A_I \equiv A(I) = a_I$, because then $v_{I-1} = (1 + \alpha''_{I-1})(1 + \alpha'_{I-1})U_{I-1}$.

Continuing, if $[BB(I)/U_{I-1}]$ does not overflow then the value stored for U_I will be

$$\begin{aligned} U_I &= [[A_I - (1 + \rho''_I)(1 + \alpha''_{I-1})(1 + \alpha'_{I-1})BB_I/v_{I-1}] - X] \\ &= ((A_I - (1 + \rho''_I)(1 + \alpha''_{I-1})(1 + \alpha'_{I-1})BB_I/v_{I-1})/(1 + \alpha'_I) \\ &\qquad - X)/(1 + \alpha''_I) \\ &\equiv v_I/((1 + \alpha''_I)(1 + \alpha'_I)) \end{aligned}$$

where the rounding errors of addition are bounded by $|\alpha'_I| < \epsilon$ and $|\alpha''_I| < \epsilon$. This result advances the induction from $n = I - 1$ to $n = I$ as desired, and lays a firm foundation for an error bound for the eigenvalues.

Let the matrix $J(X)$ be defined now to have

$A_n - \alpha'_n X$ in place of a_n, and

$\sqrt{(1 + \rho'_n)(1 + \rho''_n)(1 + \alpha''_{n-1})(1 + \alpha'_{n-1})}\, B_{n-1}$ in place of b_{n-1}.

Certainly $J(X)$ is close to J; more precisely, but neglecting terms of order ϵ^2 and η,

$$|J(X) - J| \le 2\epsilon|J - \operatorname{diag} J| + \epsilon|X|I + \sqrt{\eta}\, H$$

elementwise, where

J - diag J ≡ only the off-diagonal terms b_n in J

H ≡ the tri-diagonal matrix with all elements = 1 .

Denote the eigenvalues of J(X) by

$$\lambda_1(X) \leq \lambda_2(X) \leq \ldots \leq \lambda_N(X)$$

to correspond with the eigenvalues λ_i of J .

The Absolute Error in λ_k:

The reason for constructing $J(X)$ was that $NU(X)$ would be the number of $J(X)$'s eigenvalues $\lambda_i(X) < X$, and now this can be proved. For

$$NU(X) = \text{the number of values } U_n(X) < 0$$
$$= \text{the number of values } v_n < 0 \ ,$$

and the v_n are to $J(X)$ what the uncontaminated values u_n are to J. And each eigenvalue $\lambda_i(X)$ differs from the corresponding λ_i by no more than

$$\text{lub}_S(J(X) - J) \leq 2\epsilon \ \text{lub}_S(|J - \text{diag } J|) + \epsilon|X|$$
$$+\sqrt{\eta} \ \text{lub}_S(H) \ .$$

Here

$$\sqrt{\eta} \ \text{lub}_S(H) \leq 3\sqrt{\eta} = 3\tau(\tau\Omega) \leq 3\tau \ \text{lub}_S(J) \ ,$$

and

$$2\epsilon \ \text{lub}_S(|J - \text{diag } J|) \leq 2\epsilon \ \text{lub}_S(J)$$

by virtue of the more general form of our earlier result no. 2 with $\sqrt{\alpha^2 + \beta^2} = \sqrt{0 + (2\epsilon)^2}$. Finally, the only values of X that will concern us below are those which approximate some eigenvalue λ_i, so we can certainly assume that $\epsilon|X| \leq \epsilon \max_i|\lambda_i| = \epsilon \ \text{lub}_S(J)$ to within a negligible extra error of the order of $\epsilon^2|X|$. For those values of X we have

$$|\lambda_i(X) - \lambda_i| \leq r \cdot 3(\epsilon + \tau)\max_j|\lambda_j|$$

as a bound for the difference between the corresponding eigenvalues of $J(X)$ and of J. This means that, as X varies over the allowable arguments, each eigenvalue $\lambda_i(X)$ remains confined to some fixed interval

$$\lambda_i - r \leq \lambda_i(X) \leq \lambda_i + r \quad .$$

We have already seen that for any given k there is precisely one value X_k which, with its successor X_k', satisfies

$$NU(X_k) < k \leq NU(X_k') \quad ;$$

these values can easily be computed. And the relationship between $NU(X)$ and the $\lambda_i(X)$ tells us that

$$X_k \leq \lambda_k(X_k) \leq \lambda_k + r \qquad \text{and}$$

$$\lambda_k - r \leq \lambda_k(X_k') < X_k' \quad .$$

Since

$$0 \leq X_k' - X_k \leq 2\epsilon \max_i|\lambda_i| \quad \text{on a rounding machine}$$
$$\leq \epsilon \max_i|\lambda_i| \quad \text{on a truncating machine}$$

(we might as well assume now that arithmetic is rounded), we can accept either X_k or X_k' as an approximation to λ_k and commit an error no larger than

$$r + 2\epsilon \max_j|\lambda_j| = (5\epsilon + 3\tau)\max_j|\lambda_j| \quad .$$

This bound compares favourably with that obtained by Wilkinson (1965, p. 304-5) for the Sturm-sequence (φ-recurrence) algorithm in the absence of over/underflow. In our notation his bound is $17\epsilon \max_j|\lambda_j|$, although the use of our more refined methods reduces this to $8.75\epsilon \max_j|\lambda_j|$. This bound is not appreciably increased if Wilkinson's 1962 program is amended to cope with over/underflow, but then the φ-recurrence becomes much slower than the u-recurrence. Therefore the u-recurrence has all the advantages of speed, simplicity and accuracy over the φ-recurrence. On a computer using rounded binary floating point arithmetic, the biggest eigenvalue can be computed to within a guaranteed relative error of 2.5 units in its last place, and no eigenvalue will suffer a larger absolute error. For chopped arithmetic the guarantee is 4 units in the last place. These bounds are impressively small; but they are substantially larger than most of the errors observed in practice.

Why?

The Class $\mathcal{J}$ of neighbours of J :

A nicer appreciation of the accuracy of the u-recurrence can be achieved through the consideration of the class $\mathcal{J}$ of symmetric tri-diagonal matrices $\dot{J}$ which satisfy

$$|J - \dot{J}| \leq 2\epsilon|J - \operatorname{diag} J| \quad .$$

For example, on the rounding binary computer mentioned above the class $\mathcal{J}$ consists of those matrices $\dot{J}$ obtained from J by changing each off-diagonal element of J by at most one unit in its last place. The set $\mathcal{J}$ is a convex set in the sense that if $\dot{J}_0$ and $\dot{J}_1$ are members of $\mathcal{J}$ then so are all matrices of the form

$$t\dot{J}_1 + (1 - t)\dot{J}_0 \quad \text{for} \quad 0 \leq t \leq 1$$

lying "between" $\dot{J}_0$ and $\dot{J}_1$. Each matrix $\dot{J}$ in $\mathcal{J}$ has a set of eigenvalues

$$\dot{\lambda}_1 \leq \dot{\lambda}_2 \leq \cdots \leq \dot{\lambda}_{N-1} \leq \dot{\lambda}_N \quad ;$$

and as $\dot{J}$ varies over $\mathcal{J}$ each eigenvalue $\dot{\lambda}_k$ varies over some set Λ_k which can also be shown to be a closed convex set. In other words, associated with the class $\mathcal{J}$ of matrices $\dot{J}$ is the set of N intervals

$$\Lambda_k = \text{the set of all } x = \dot{\lambda}_k \text{ for some } \dot{J} \text{ in } \mathcal{J} \quad .$$

Some of these intervals may overlap, but it is soon seen that no Λ_k can be contained strictly inside another. Therefore the intervals Λ_k share the same ordering as the eigenvalues λ_k. Obviously Λ_k is contained in

$$\lambda_k - 2\epsilon \max_j|\lambda_j| \leq x \leq \lambda_k + 2\epsilon \max_j|\lambda_j| \quad ,$$

but the interval Λ_k hardly ever occupies more than a small fraction of that latter interval.

The significance of the interval Λ_k is that for most practical purposes any number in Λ_k is as acceptable an approximation to λ_k as any other. Such might be the case, for example, if each off-diagonal element b_i of J were independently in error by as much as $2\epsilon|b_i|$ because of previous rounding errors. The independence of the errors is essential; correlations among the errors in the b_i could conceivably cause the eigenvalues of J not to be in error at all, as would be the case if

$$\begin{pmatrix} 0 & 1+\epsilon & 0 \\ 1+\epsilon & 0 & 1-\epsilon \\ 0 & 1-\epsilon & 0 \end{pmatrix} \quad \text{were erroneously computed as} \quad \begin{pmatrix} 0 & 1-\epsilon & 0 \\ 1-\epsilon & 0 & 1+\epsilon \\ 0 & 1+\epsilon & 0 \end{pmatrix} .$$

As long as the errors in the b_i are independent, the width of the interval Λ_k is an indication of the extent to which λ_k must be regarded as inescapably uncertain. And that the error introduced by the programs analyzed here contributes negligibly to this inescapable uncertainty shall now be demonstrated.

Let $\dot{J}(X)$ be defined to have elements $\dot{a}_n = a_n$ and $\dot{b}_{n-1} = \sqrt{(1+\beta'_n)(1+\beta''_n)(1+\alpha''_{n-1})(1+\alpha'_n)}\ b_{n-1}$, where the Greek letters were defined during the construction of $J(X)$. Except for terms of order ϵ^2 which shall be ignored,

$$|\dot{b}_{n-1} - b_{n-1}| \leq 2\epsilon|b_{n-1}|$$

so $\dot{J}(X)$ belongs to $\mathcal{J}$ and each eigenvalue $\dot{\lambda}_k(X)$ of $\dot{J}(X)$ lies in its corresponding interval Λ_k. Also,

$$|\dot{J}(X) - J(X)| \leq \epsilon|X|I + \sqrt{\eta}\ H$$

except for ignorable terms of order η, so

$$|\dot{\lambda}_k(X) - \lambda_k(X)| \leq r(X) = \epsilon|X| + 3\tau \max_i|\lambda_i| \quad .$$

And if X_k and its successor X'_k are defined, as before, by

$$NU(X_k) < k \leq NU(X'_k)$$

then

$$X_k \leq \lambda_k(X_k) \leq \dot{\lambda}_k(X_k) + r(X_k) \quad \text{and}$$

$$\dot{\lambda}_k(X'_k) - r(X'_k) \leq \lambda_k(X'_k) \leq X'_k \quad .$$

If arithmetic is rounded, then

$$X'_k - X_k \leq 2\epsilon \max.(|X_k|\ ,\ |X'_k|)$$

(except possibly when $X'_k - X_k = \eta$, which is ignored). Putting all the facts together shows that either X_k or X'_k differs from $\lambda'(X'_k)$ or $\lambda'(X_k)$ respectively by no more than

$$2\epsilon \max.(|X_k|, |X'_k|) + 3\tau \max_i |\lambda_i| .$$

On a binary machine with rounded floating point arithmetic we may summarize this result as follows:

The computed approximation to the k^{th} eigenvalue λ_k of J need not differ by more than $1\frac{1}{2}$ units in its last place from the k^{th} eigenvalue λ'_k of some matrix J'_k each element of which differs from the corresponding element of J by at most one unit in its last place, plus an absolute error of

$$3\tau \max_i |\lambda_i|$$

from over/underflow subterfuges.

In other words, if J is already uncertain in each element by several units in its last place, and if

$$|\lambda_k| / \max_i |\lambda_i| > 3\tau/\epsilon$$

(which is not often a restriction since $3\tau/\epsilon < 10^{-10}$ on each of the machines tabulated, even in double-precision), then the additional uncertainty introduced by the computation of λ_k will be insignificant when compared with the intrinsic uncertainty in λ_k caused by uncertainty in J. If λ_k

is intrinsically uncertain by only a few units in its last place, then the approximation to λ_k will be accurate to within a few units in its last place too despite the fact that λ_k is much smaller than $\max_i |\lambda_i|$. This partially explains why some of the very small eigenvalues of symmetric tridiagonal matrices have been computed to such unexpectedly fine relative precision by the u-recurrence.

Insensitive Eigenvalues:

But why are the smaller eigenvalues of J so frequently (but not always) so much less sensitive to small relative perturbations in J than might be suggested by simple examples like the following?

$$J = \begin{pmatrix} 1 & 1-\epsilon \\ 1-\epsilon & 1 \end{pmatrix} , \quad \lambda_1 = \epsilon , \quad \lambda_2 = 2-\epsilon \quad .$$

Unlike this example are many others where even the tiniest eigenvalues suffer relative (rather than absolute) displacements which are of a comparable order of magnitude with the relative changes in the off-diagonal elements of J. (Although not always easy to explain, it is often observed that J's eigenvalues are less sensitive to relative perturbations in the off-diagonal elements than to comparable relative perturbations in the diagonal elements.) An extreme example of this phenomenon is provided by those matrices J whose diagonal elements all vanish. These matrices turn up during certain computations of singular values; see Golub and Kahan (1965) p. 213. The methods used above can be exploited to prove that

If J is an $N \times N$ symmetric tri-diagonal matrix,

if $\quad \operatorname{diag} J = 0 , \quad$ and if $\quad |\delta J| < \epsilon |J| \quad ,$

then the ordered eigenvalues λ_i of J and $\lambda_i + \delta\lambda_i$ of $J + \delta J$ satisfy

$$|\delta\lambda_i| \leq N\epsilon |\lambda_i| / (1 - N\epsilon) ,$$

provided $N\epsilon < 1$.

An outline of the proof follows. Write

$$b_i + \delta b_i = b_i(1 + \beta_i) \quad \text{with} \quad |\beta_i| < \epsilon \ .$$

Without loss of generality we may assume

$$b_i \neq 0 \quad \text{for} \quad 1 \leq i < N \ .$$

Corresponding to the u-recurrence applied to $J - xI$ is the corresponding v-recurrence, say, that belongs to $(J + \delta J) - xI$; they can best be compared when written side by side thus:

$$u_1 = -x \ , \qquad\qquad v_1 = -x \ ,$$

$$u_i = -x - b_{i-1}^2/u_{i-1} \ , \qquad\qquad v_i = -x - (1 + \beta_{i-1})^2 b_{i-1}^2/v_{i-1} \ .$$

It is well known and easy to show that both J and $J + \delta J$ have only simple eigenvalues λ_k and $\lambda_k + \delta\lambda_k$ respectively, and that $u_N = 0$ if and only if x is an eigenvalue of J , and that $v_N = 0$ if and only if x is an eigenvalue of $J + \delta J$. (cf. Wilkinson (1965) p. 300.)

Our object now is to show that each λ_k is the k^{th} eigenvalue of some matrix which differs from $J + \delta J$ by terms of order $\epsilon\lambda_k$ rather than $\epsilon|J|$. There are two cases according as N is odd or even.

If $N = 2n-1$ we define the factors $(1 + \gamma_i)$ via

$$\begin{aligned} 1 + \gamma_n &= 1 \\ 1 + \gamma_{i+1} &= (1 + \beta_i)^2/(1 + \gamma_i) \quad \text{if} \quad i \geq n \\ 1 + \gamma_{i-1} &= (1 + \beta_{i-1})^2/(1 + \gamma_i) \quad \text{if} \quad i \leq n \ . \end{aligned}$$

$$w_i \equiv (1 + \gamma_i)u_i \quad \text{and} \quad x_i \equiv -\gamma_i x \quad ,$$

and observe that

$$w_1 = x_1 - x$$

$$w_i = x_i - x - (1 + \beta_{i-1})b_{i-1}^2/w_{i-1} \quad .$$

This is just the w-recurrence, say, belonging to the matrix

$$J(x) - xI = J + \delta J + \mathrm{diag}(x_i) - xI \quad .$$

Since $$|J + \delta J - J(x)| = |x|\mathrm{diag}|\gamma_i|$$

$$\leq |x| \ \mathrm{diag}[(1 - \epsilon)^{-2n+2} - 1]$$

$$\leq N\epsilon|x|I/(1 - N\epsilon)$$

if $N\epsilon < 1$, the k^{th} eigenvalue $\lambda_k + \delta\lambda_k$ of $J + \delta J$ differs from the k^{th} eigenvalue of $J(x)$ by no more than $N\epsilon|x|/(1 - N\epsilon)$. But the k^{th} eigenvalue of $J(\lambda_k)$ is just λ_k since $\mathrm{sign}(w_i) = \mathrm{sign}(u_i)$ for all x and $w_N = u_N = 0$ for $x = \lambda_k$. Therefore

$$|\lambda_k + \delta\lambda_k - \lambda_k| \leq N\epsilon|\lambda_k|/(1 - N\epsilon)$$

as claimed.

A similar schema works when N is even. Thus, one can hardly be surprised in this case when each computed eigenvalue of such a matrix J is correct to within N units in its last place despite a wide variation in the orders of magnitudes of the eigenvalues.

The possible persistence of high relative precision in many of the tiny eigenvalues of wider classes of matrices J awaits a systematic explanation with predictive powers, in the absence of which it is hard to say when a small computed eigenvalue has higher relative precision than is implied by the absolute error bound

$$(5\epsilon + 3\tau)\max_j |\lambda_j| \quad .$$

Conclusion:

There are faster programs than those described here, but none more elegant nor more accurate.

Acknowledgements:

This work was done while the author enjoyed six months at Stanford University on leave of absence from the University of Toronto. The author thanks Professor G. E. Forsythe and Dr. J. Varah for their help and encouragement.

References:

F. R. Gantmacher (1959) "The Theory of Matrices," 2 vol., translated by K.A. Hirsch for Chelsea, New York.

G. Golub and W. Kahan (1965) "Calculating the Singular values and Pseudo-Inverse of a Matrix" J. SIAM Numer. Anal. Ser. B, 2 p. 205-223.

A. S. Householder (1964) "The Theory of Matrices in Numerical Analysis" Blaisdell, New York.

W. Kahan and J. Varah (1966) "Two working Algorithms for the Eigenvalues of a symmetric tridiagonal Matrix", CS43, Computer Science Dept.Stanford,Calif.

Betty Jane Stone (1962) "Best possible ratios of matrix norms" Numer. Math. 4 p. 114-6. For more detail, see the Stanford University Applied Mathematics and Statistics Lab. tech. report no. 19 (May 10, 1962) of almost the same name.

J. H. Wilkinson (1962) "Calculation of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection. Numer. Math. 4 p. 362-367.

J. H. Wilkinson (1963) "Rounding Errors in Algebraic Processes " Notes on Appl. Sci. no. 32, HMSO, London.

J. H. Wilkinson (1965) "The Algebraic Eigenproblem" Oxford U.P.

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Computer Science Department Stanford University Stanford, California 94305	2a. REPORT SECURITY CLASSIFICATION Unclass. 2b. GROUP ---

3. REPORT TITLE

ACCURATE EIGENVALUES OF A SYMMETRIC TRI-DIAGONAL MATRIX

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Manuscript for Publication (Technical Report)

5. AUTHOR(S) (Last name, first name, initial)

KAHAN, W

6. REPORT DATE July 22, 1966	7a. TOTAL NO. OF PAGES 52	7b. NO. OF REFS 8
8a. CONTRACT OR GRANT NO. Nonr-225(37) b. PROJECT NO. NR-044-211 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) CS41 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) none	

10. AVAILABILITY/LIMITATION NOTICES

Releasable without limitations on dissemination.

11. SUPPLEMENTARY NOTES ---	12. SPONSORING MILITARY ACTIVITY Office of Naval Research Code 432 Washington, D. C. 20360

13. ABSTRACT

Having established tight bounds for the quotient of two different lub-norms of the same tri-diagonal matrix J, the author observes that these bounds could be of use in an error-analysis provided a suitable algorithm were found. Such an algorithm is exhibited, and its errors are thoroughly accounted for, including the effects of scaling, over/under-flow and roundoff. A typical result is that, on a computer using rounded floating point binary arithmetic, the biggest eigenvalue of J can be computed easily to within 2.5 units in its last place, and the smaller eigenvalues will suffer absolute errors which are no larger. These results are somewhat stronger than had been known before.

DD FORM 1 JAN 64 1473

Unclassified

Security Classification

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
1. Tri-Diagonal Symmetric Matrix.						
2. Eigenvalue error-analysis.						
3. Ration of lub-norms.						

END

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)*, *(S)*, *(C)*, or *(U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, roles, and weights is optional.

DD FORM 1 JAN 64 1473 (BACK)

Unclassified

Security Classification